Complex systems analysis of EEG data - Module 5

Ethel Pruss, Anita Vrins, Jos Prinsen, Caterina Ceccato

3/17/2023

Applying attractor dynamics methods to EEG data collected from a storytelling task involving a social robot

Dataset

The dataset used was collected by the authors in collaboration with another student during experiments conducted at Tilburg University in fall 2022. The dataset contains EEG data from 40 participants, divided in two conditions. It was a within-participant study, so for each participant data from both conditions are available. Every datafile includes EEG data from 8 channels (Fz, C3, Cz, C4, Pz, PO7, Oz, PO8) collected from a 10 minutes interaction. The conditions refer to the experimental setup of the study, and will not be discussed in depth in here.

Preprocessing steps applied prior to the following analysis

The data was collected through Matlab Simulink, therefore it was initially stored in .mat files. The files have been first converted into .csv and then to .fif to be able to work with the mne module for python. After a first inspection of the data, we noticed that it was very noisy, and we decided it would be better to preprocess it through EEGLAB, to make sure we could get a clear idea of the steps that were applied. Our preprocessing pipeline consisted in

1. Loading the data (alredy notch-filtered at 50 Hz)
2. Applying a bandpass filter of 0.5-30 Hz. We decided to cut out the higher frequencies because muscular movement is often highly reflected in gamma.
3. Changing the sampling rate. This was necessary for the next step. The sampling rate of the headset used was 250, which was changed to 256.
4. Applying ASR over all channels to reject data. ASR is an effective method for removing artifacts from EEG data. It uses principal component analysis (PCA) to identify and reject data contaminated by artifacts such as eye blinks, eye movements, and muscle activity. During this step, some channels were removed and then automatically reconstructed.

Power spectral density

After these steps, we continued our analysis of the data in python. We calculated power spectral density applying the welch method in order to extract data from the aplha, beta and theta power bands for each participant. These powerbands were needed to calculate the EEG engagement index, which we have used as measure of engagement in the experiments. The data about the EEG engagement index over time was then stored in separate csv files (one for each participant, for each condition) in order to be used for further analysis in R.

Loading the data

The following analysis methods will be demonstrated on one participant out of the dataset. To adjust it to other participants, only the loaded file needs to be changed. In the example code, we will be looking at participant number 10 in the adaptive condition.

```
filename <- read.csv("data/P10Engagement_Raw.csv")
```

Loading libraries

```
library(Hmisc)
library(ggplot2)
library(grid)
library(randtests)
library(tseries)
```

Frontal EEG data

```
filename$Frontal_lag1 <- Lag(filename$Frontal, 1)
filename$Frontal_lead1 <- Lag(filename$Frontal, -1)
filename$Frontal_change1 <- (filename$Frontal_lead1 - filename$Frontal_lag1)/2
```

Cenrtal EEG data

```
filename$Central_lag1 <- Lag(filename$Central, 1)
filename$Central_lead1 <- Lag(filename$Central, -1)
filename$Central_change1 <- (filename$Central_lead1 - filename$Central_lag1)/2
```

Parietal EEG data

```
filename$Parietal_lag1 <- Lag(filename$Parietal, 1)
filename$Parietal_lead1 <- Lag(filename$Parietal, -1)
filename$Parietal_change1 <- (filename$Parietal_lead1 - filename$Parietal_lag1)/2
```

EEG data from all regions

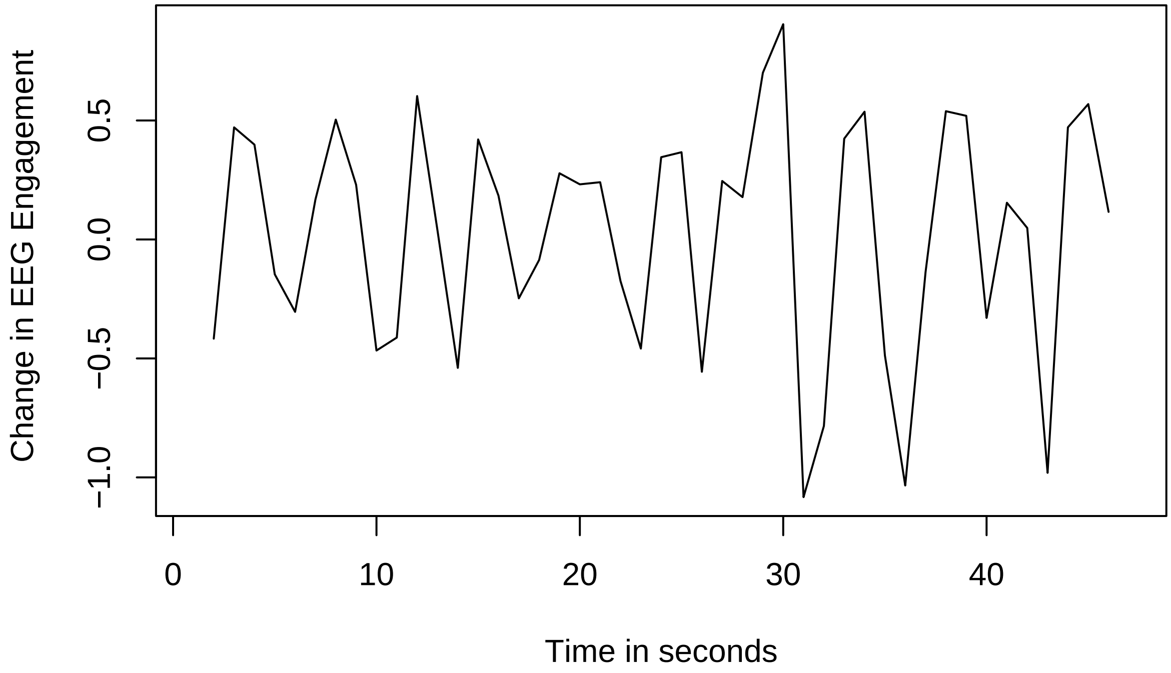
```
filename$Total_lag1 <- Lag(filename$Total, 1)
filename$Total_lead1 <- Lag(filename$Total, -1)
filename$Total_change1 <- (filename$Total_lead1 - filename$Total_lag1)/2
```

Creating time series variables

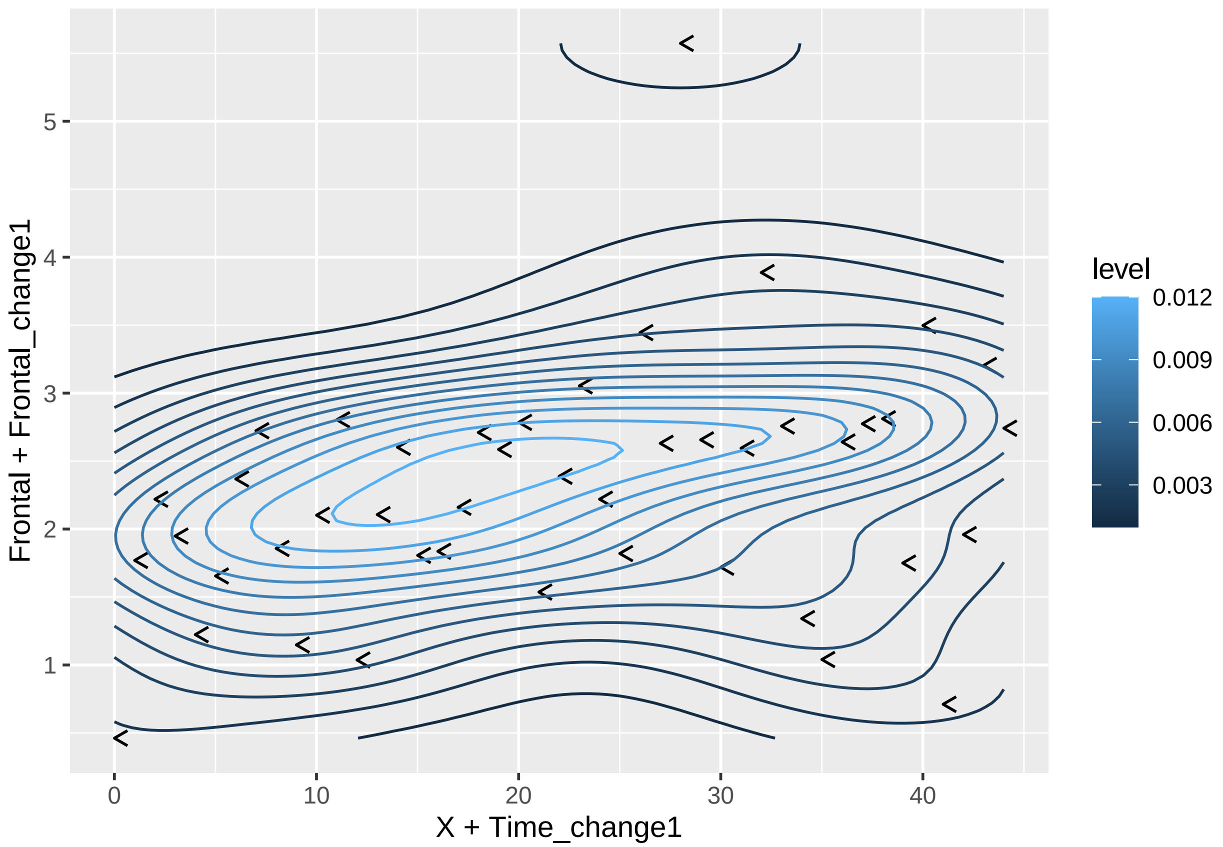
```
filename$Time_lag1 <- Lag(filename$X, 1)
filename$Time_lead1 <- Lag(filename$X, -1)
filename$Time_change1 <- (filename$Time_lag1- filename$Time_lead1)/2
```

Plotting the frontal EEG data

```
plot(filename$Frontal_change1,type='l', xlab = "Time in seconds", ylab =" Change in EEG Engagement")
```

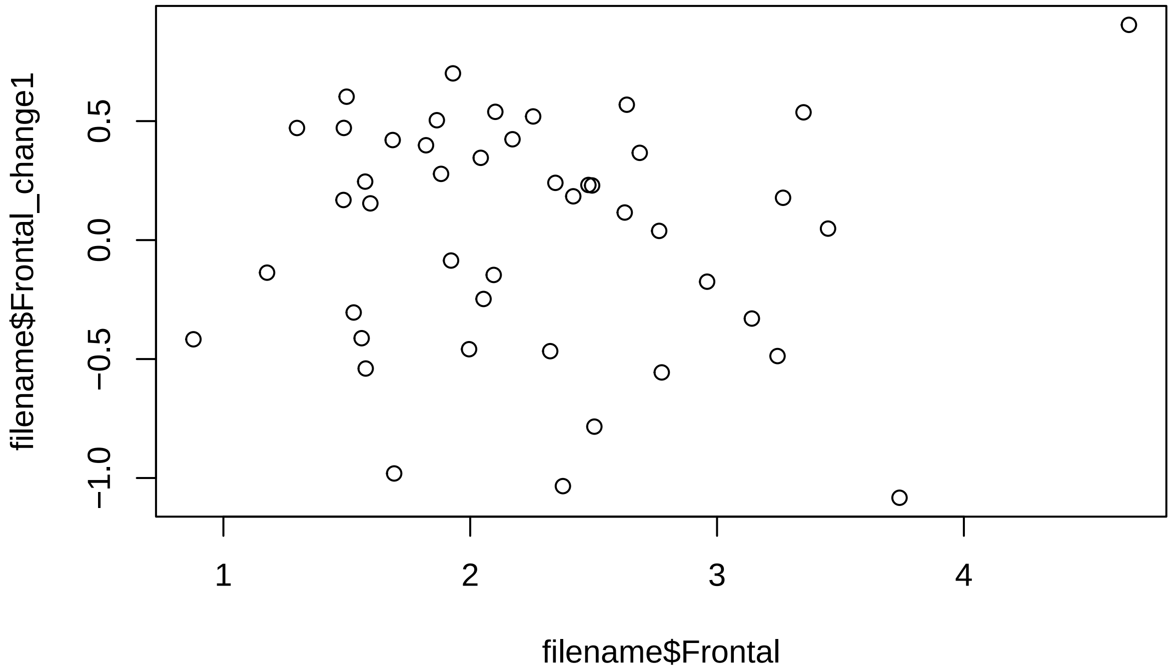


```
ggplot(data=filename,aes(x=X + Time_change1,y=Frontal + Frontal_change1))+
  geom_segment(aes(xend=X + Time_change1,yend=Frontal + Frontal_change1),arrow=arrow(length=unit(.2,"cm
  stat_density2d(aes(colour=..level..))+
  labs(list(title="Participant 1 Vector Density Plot", x= "Time", y = "EEG Engagement"))
```

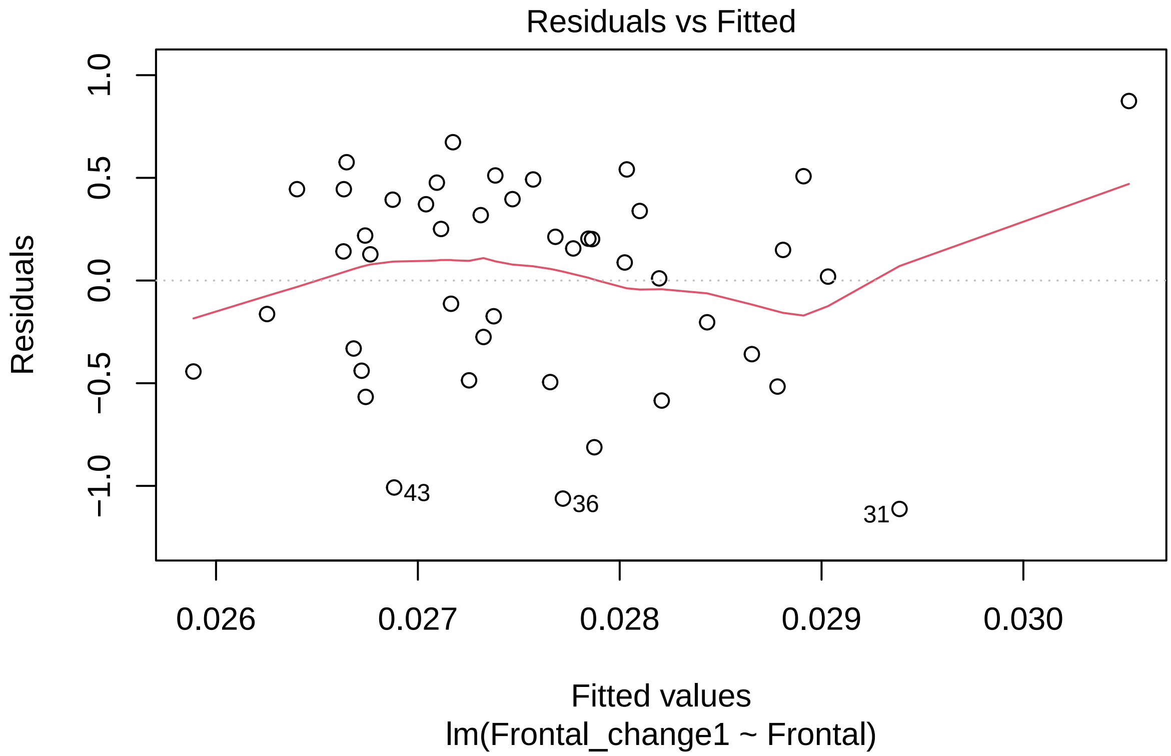


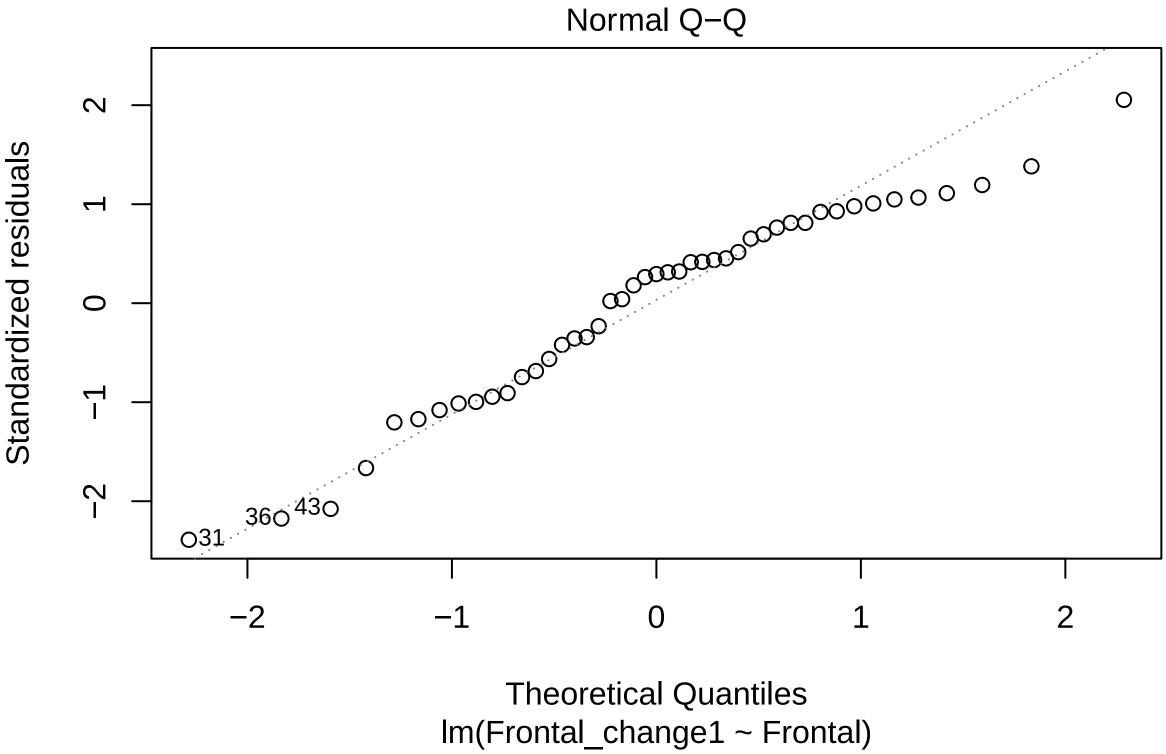
Plotting a linear predictor for the frontal data

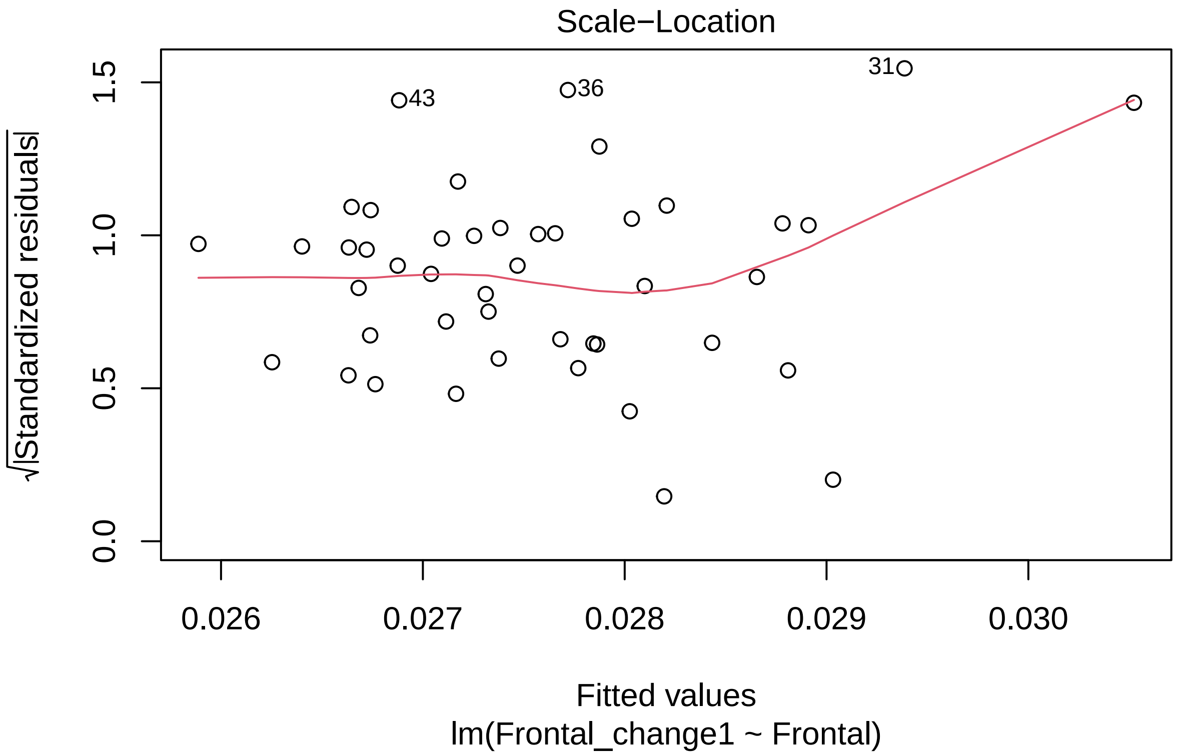
```
plot(y = filename$Frontal_change1, x = filename$Frontal)
```

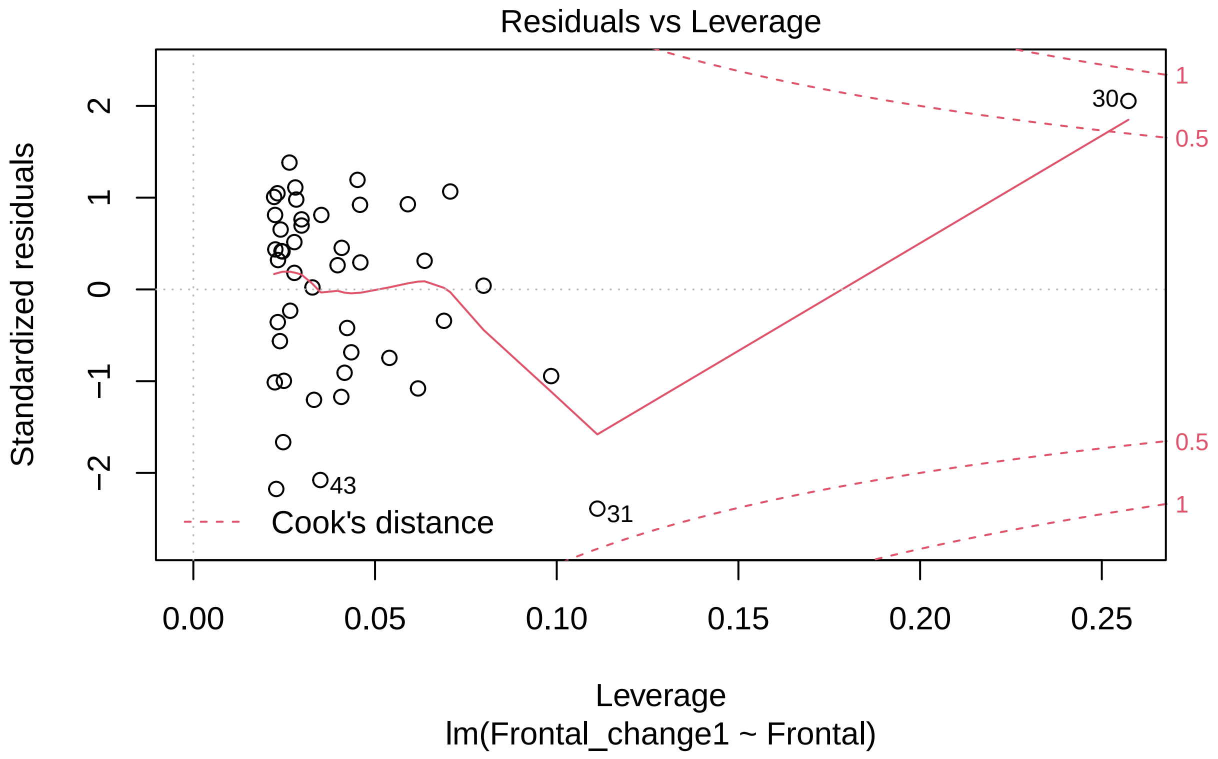


```
model1 <- lm(formula = Frontal_change1 ~ Frontal, data = filename )
plot(model1)
```


Residuals vs Fitted
Residuals
1.0
0.5
0.0
−0.5
−1.0
0.026
0.027
0.028
0.029
0.030
Fitted values
lm(Frontal_change1 ~ Frontal)
43
36
31


Normal Q−Q
Standardized residuals
2
1
0
−1
−2
31
36
43
−2
−1
0
1
2
Theoretical Quantiles
lm(Frontal_change1 ~ Frontal)


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
43
36
31
0.026
0.027
0.028
0.029
0.030
Fitted values
lm(Frontal_change1 ~ Frontal)

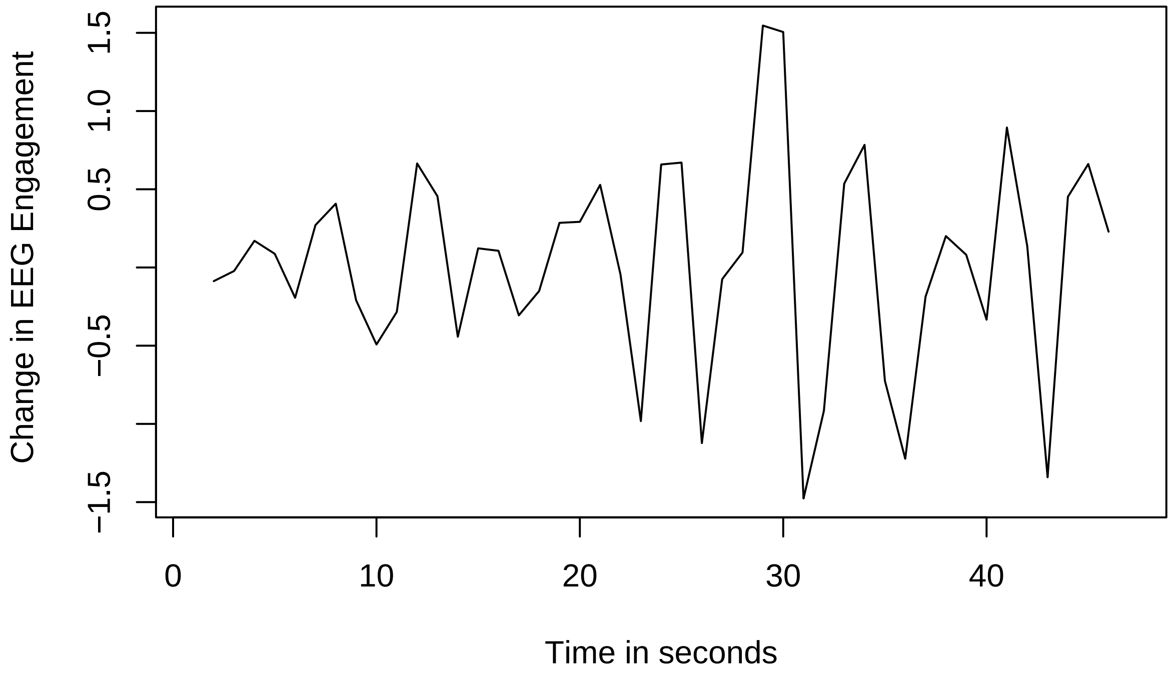


```
summary(model1)
```

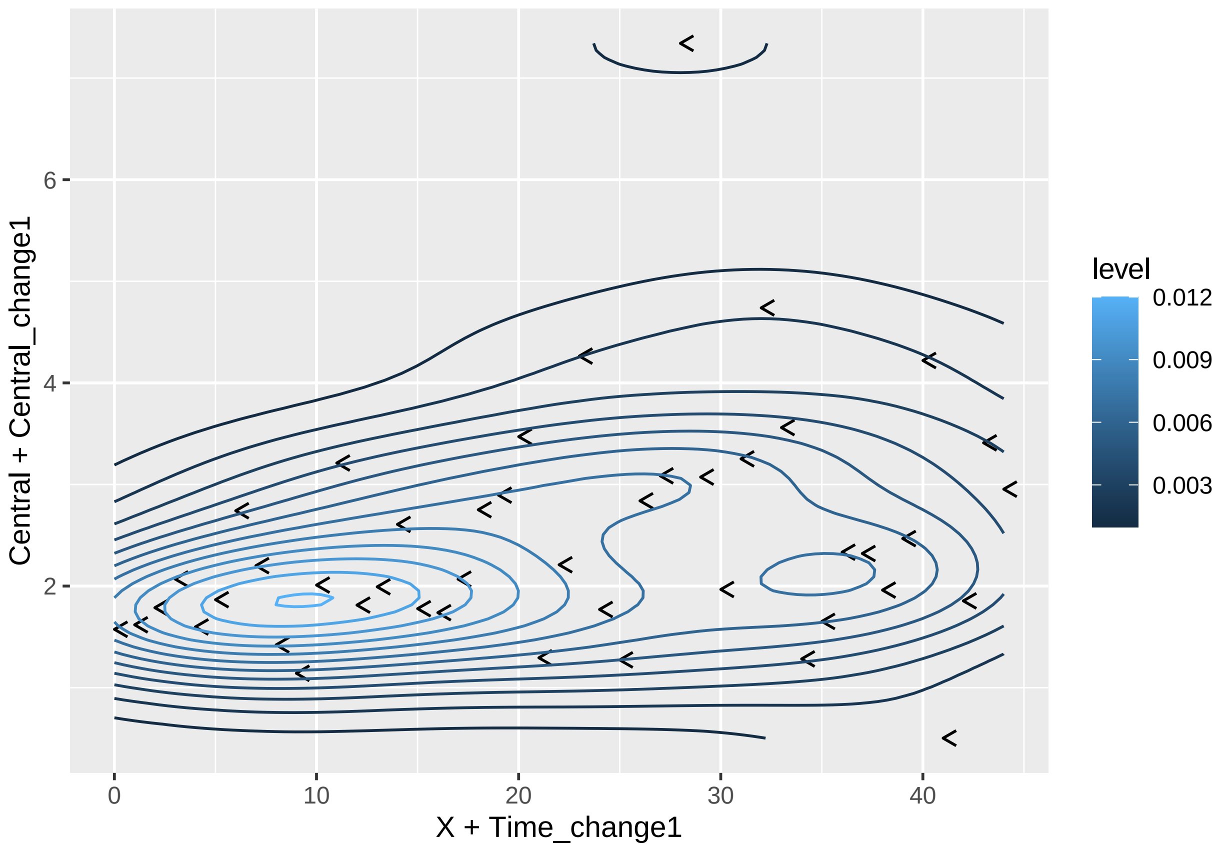
```
##
## Call:
## lm(formula = Frontal_change1 ~ Frontal, data = filename)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1123 -0.3584  0.1418  0.3936  0.8741
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.024813   0.235273   0.105    0.916
## Frontal     0.001223   0.099147   0.012    0.990
##
## Residual standard error: 0.4938 on 43 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  3.538e-06,  Adjusted R-squared:  -0.02325
## F-statistic: 0.0001521 on 1 and 43 DF,  p-value: 0.9902
```

Plotting the central EEG data

```
plot(filename$Central_change1,type='l', xlab = "Time in seconds", ylab =" Change in EEG Engagement")
```

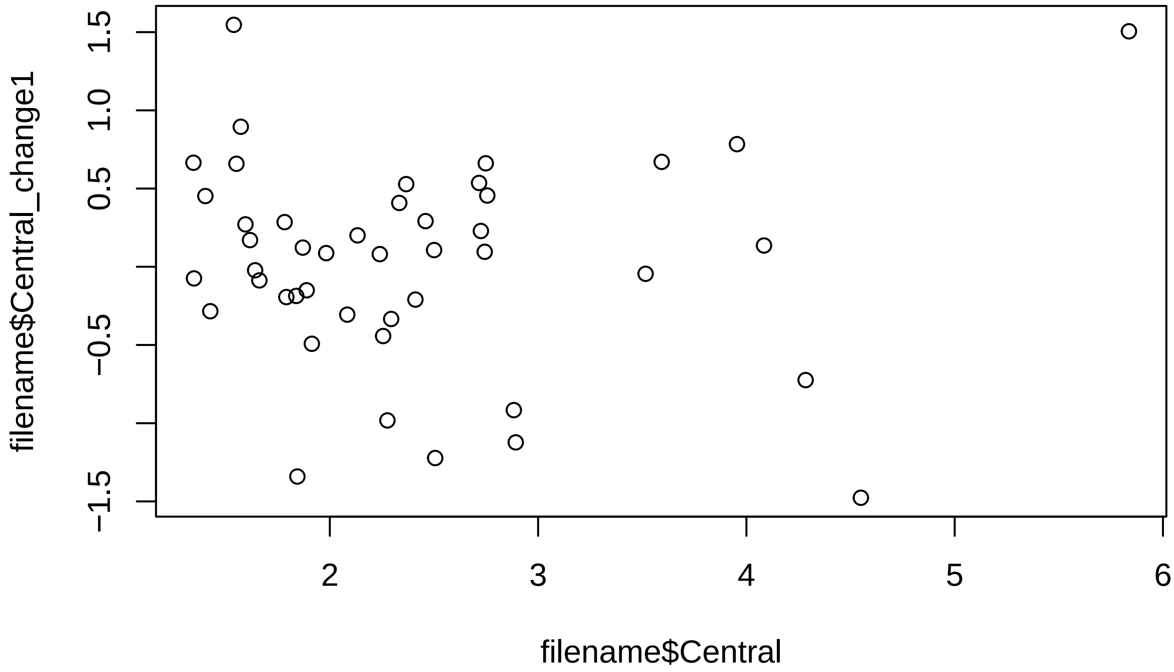


```
ggplot(data=filename,aes(x=X + Time_change1,y=Central + Central_change1))+
  geom_segment(aes(xend=X + Time_change1,yend=Central + Central_change1),arrow=arrow(length=unit(.2,"cm
  stat_density2d(aes(colour=..level..))+
  labs(list(title="Participant 1 Vector Density Plot", x= "Time", y = "EEG Engagement"))
```

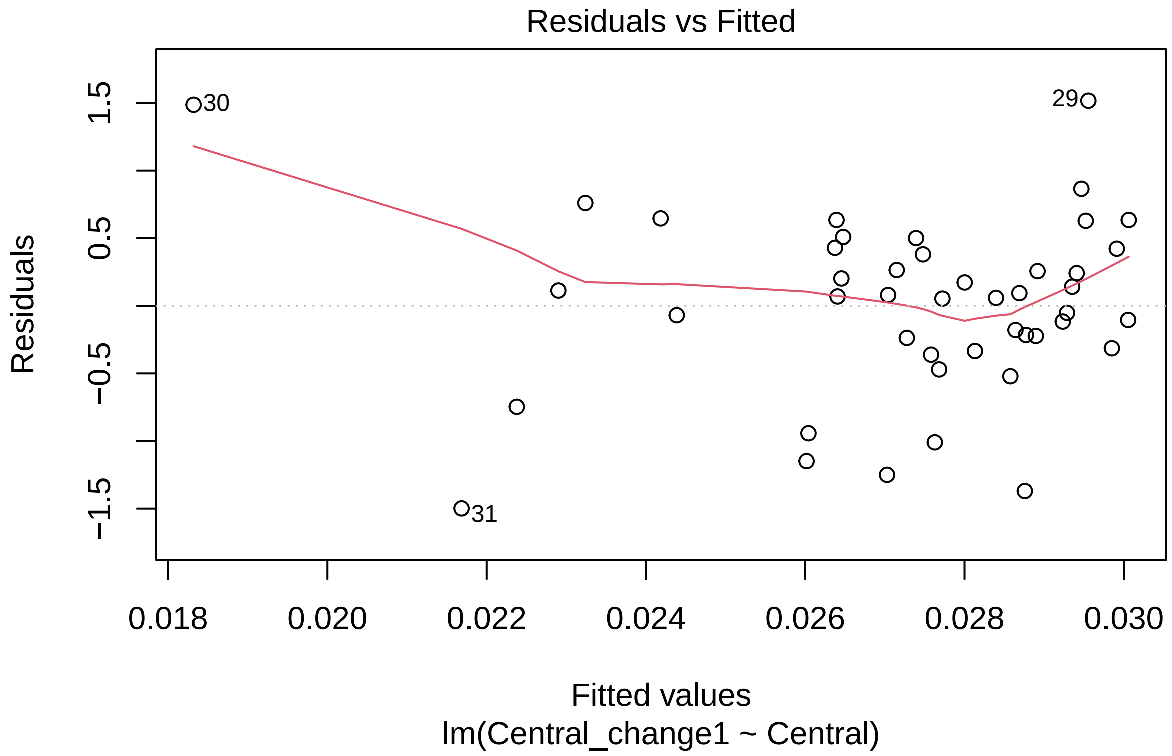


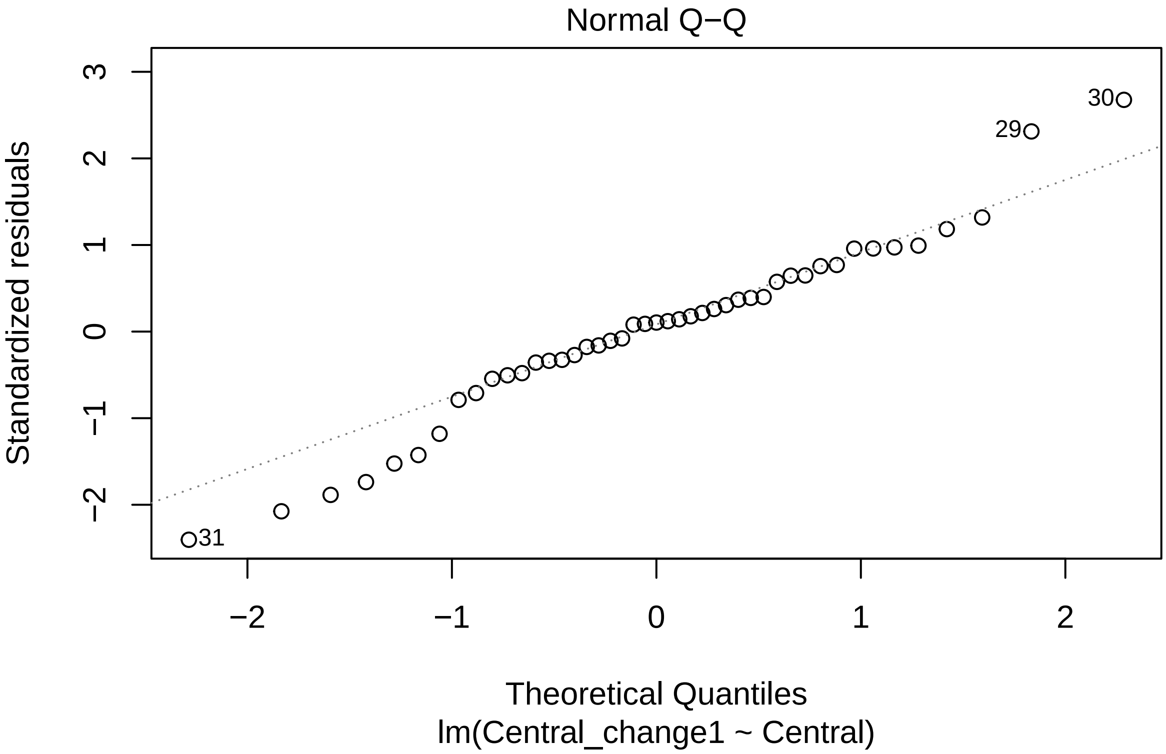
Plotting a linear predictor for the central data

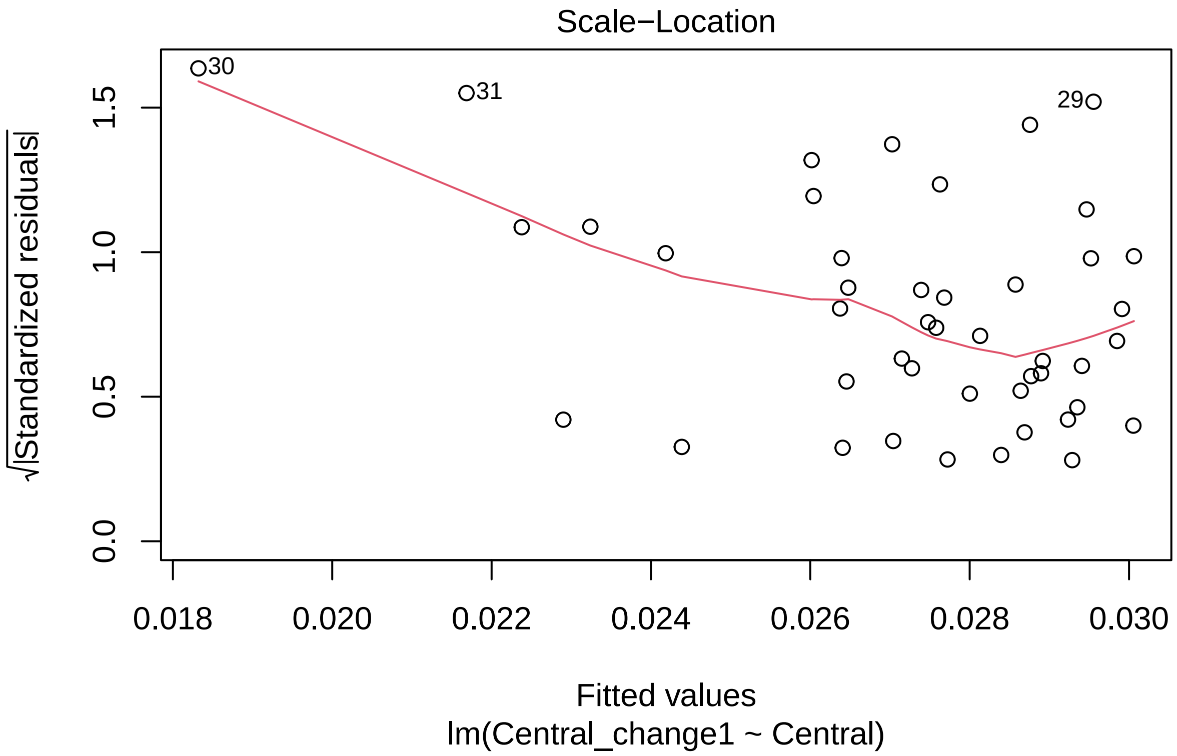
```
plot(y = filename$Central_change1, x = filename$Central)
```

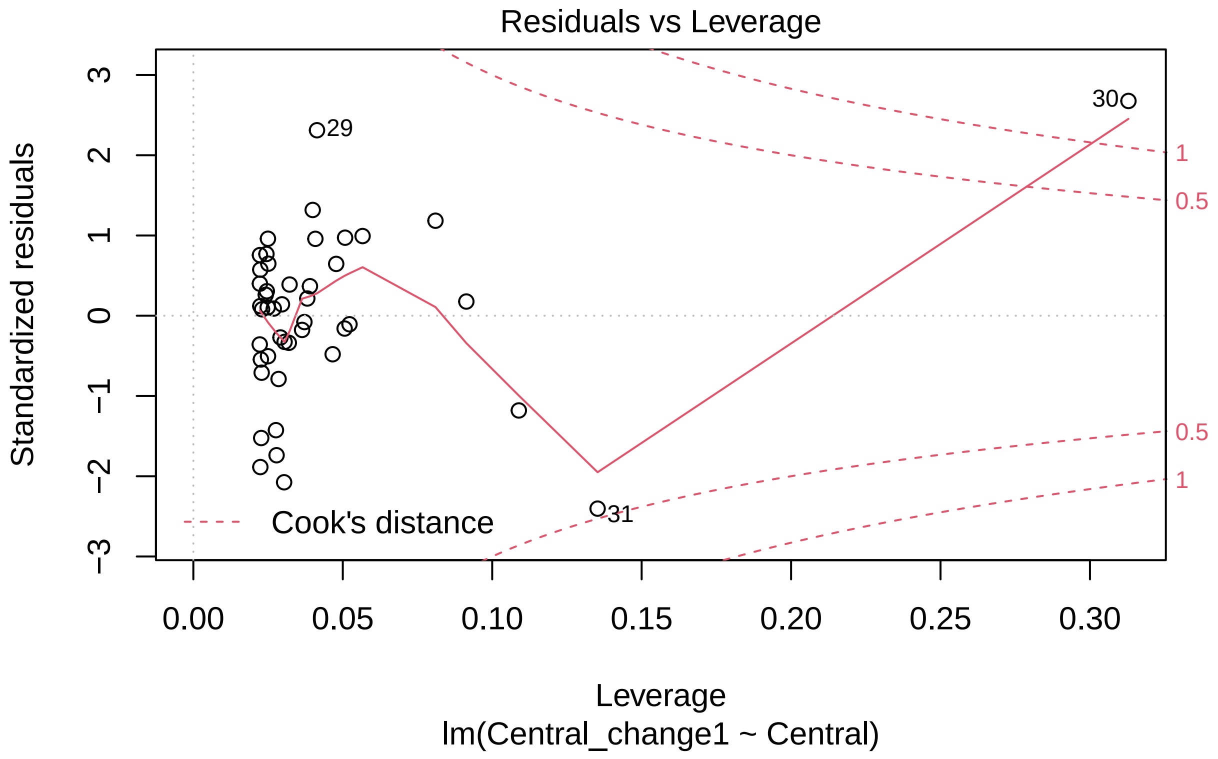


```
model1 <- lm(formula = Central_change1 ~ Central, data = filename )
plot(model1)
```


Residuals vs Fitted
Residuals
1.5
0.5
−0.5
−1.5
30
29
31
0.018
0.020
0.022
0.024
0.026
0.028
0.030
Fitted values
lm(Central_change1 ~ Central)


Normal Q−Q
Standardized residuals
3
2
1
0
−1
−2
30
29
31
−2
−1
0
1
2
Theoretical Quantiles
lm(Central_change1 ~ Central)


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
0.018
0.020
0.022
0.024
0.026
0.028
0.030
30
31
29
Fitted values
lm(Central_change1 ~ Central)

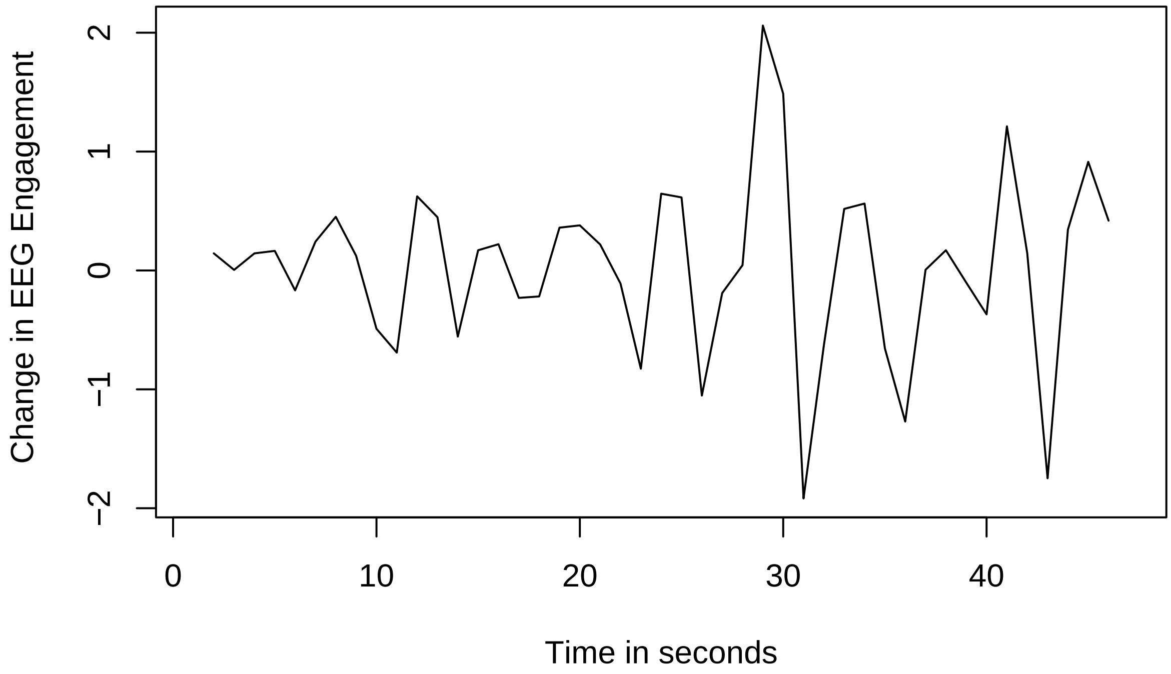


```
summary(model1)
```

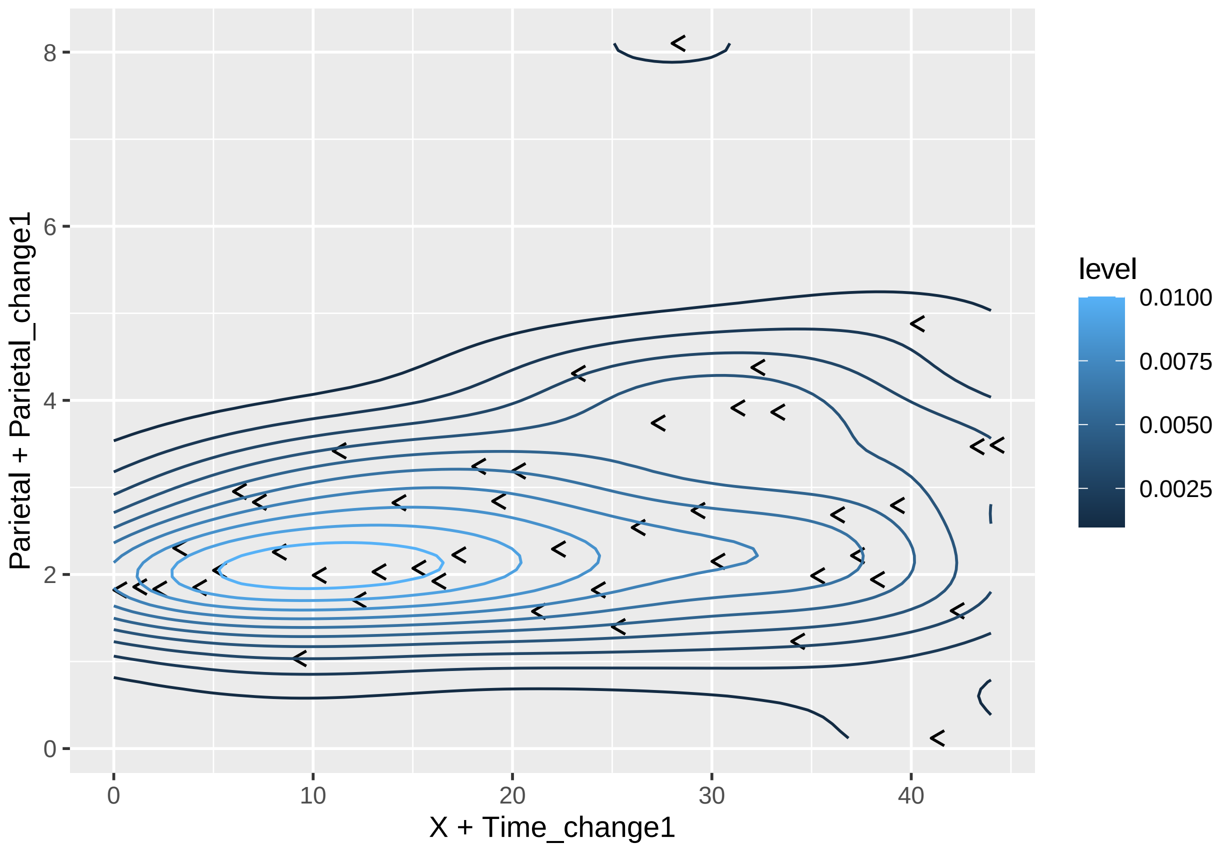
```
##
## Call:
## lm(formula = Central_change1 ~ Central, data = filename)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.49831 -0.31412  0.06916  0.42225  1.51695
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.033577   0.274232   0.122    0.903
## Central     -0.002614   0.105670  -0.025    0.980
##
## Residual standard error: 0.6703 on 43 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  1.423e-05,  Adjusted R-squared:  -0.02324
## F-statistic: 0.000612 on 1 and 43 DF,  p-value: 0.9804
```

Plotting the parietal EEG data

```
plot(filename$Parietal_change1,type='l', xlab = "Time in seconds", ylab =" Change in EEG Engagement")
```

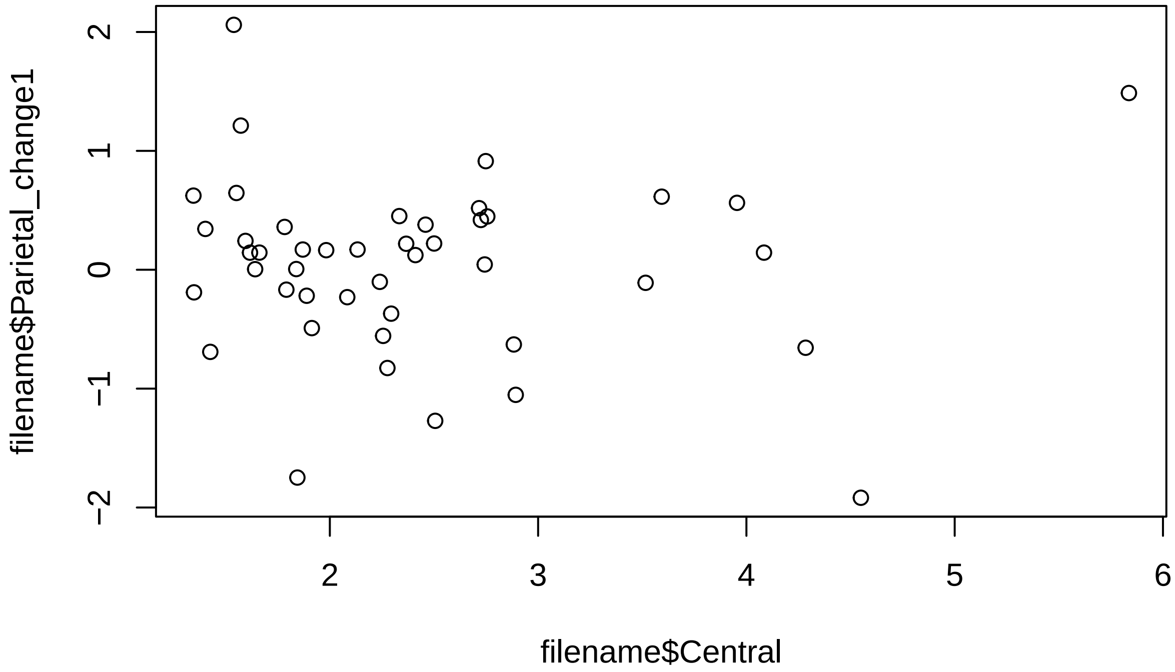


```
ggplot(data=filename,aes(x=X + Time_change1,y=Parietal + Parietal_change1))+
  geom_segment(aes(xend=X + Time_change1,yend=Parietal + Parietal_change1),arrow=arrow(length=unit(.2,"
  stat_density2d(aes(colour=..level..))+
  labs(list(title="Participant 1 Vector Density Plot", x= "Time", y = "EEG Engagement"))
```

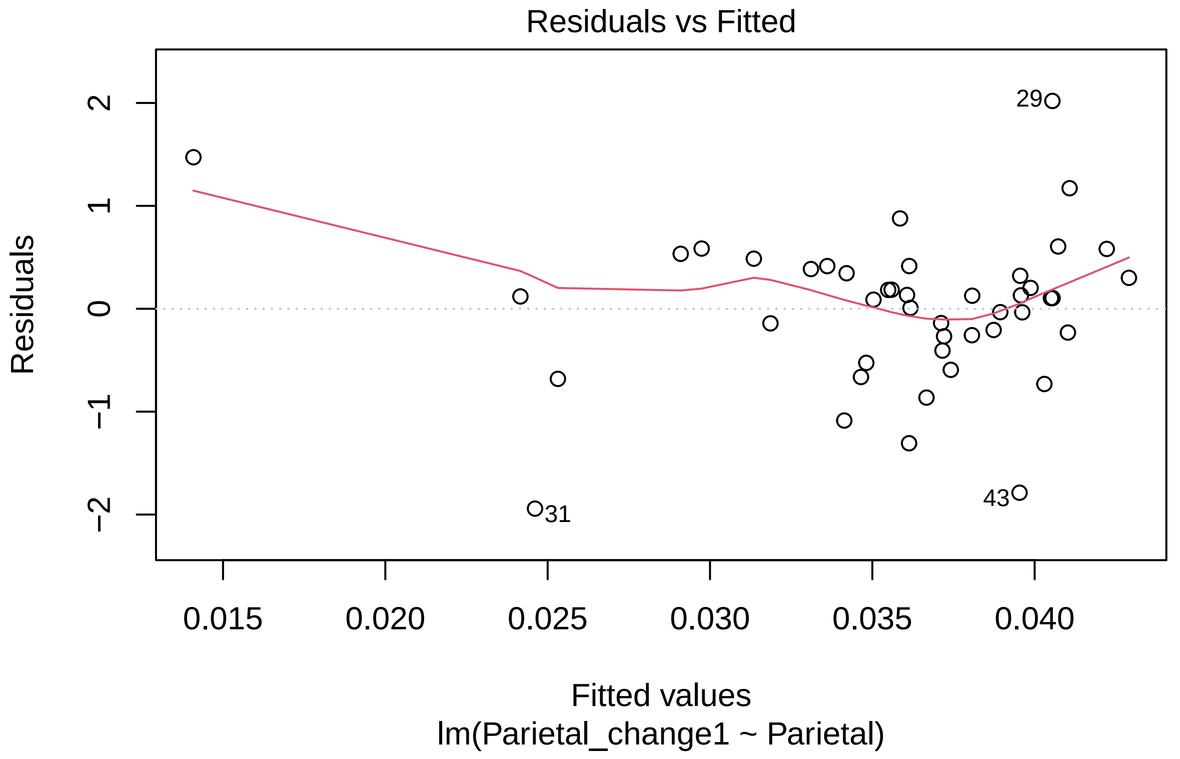


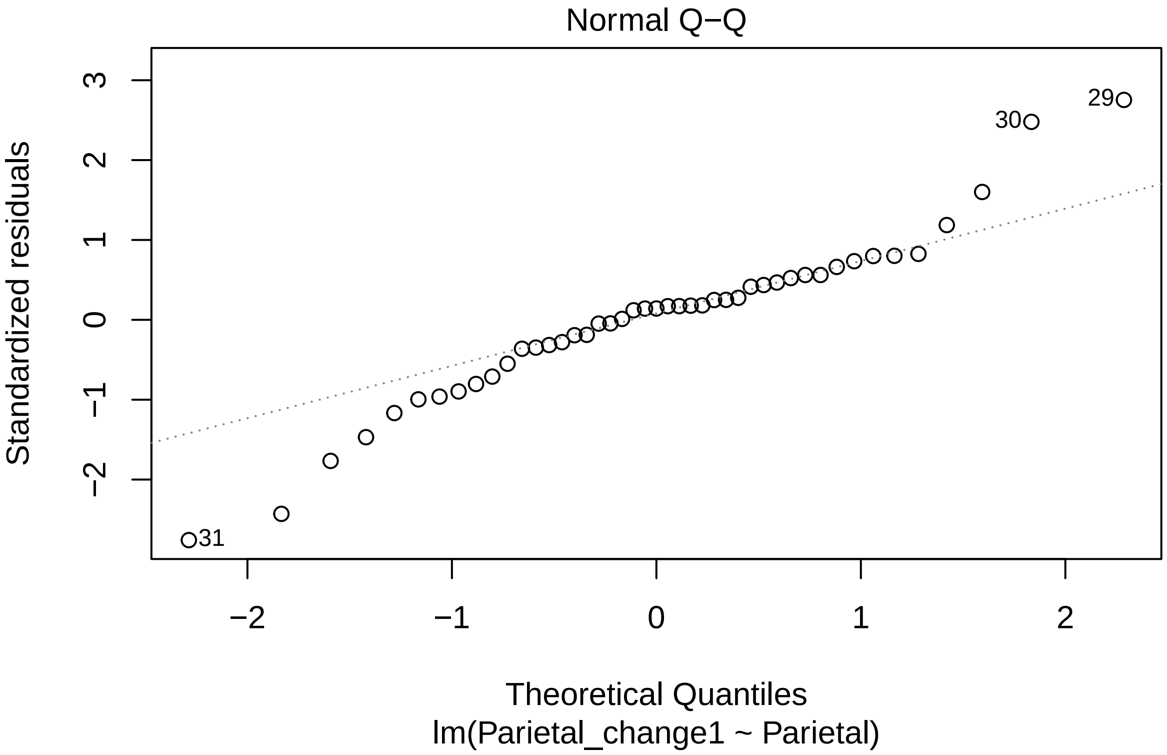
Plotting a linear predictor for the parietal data

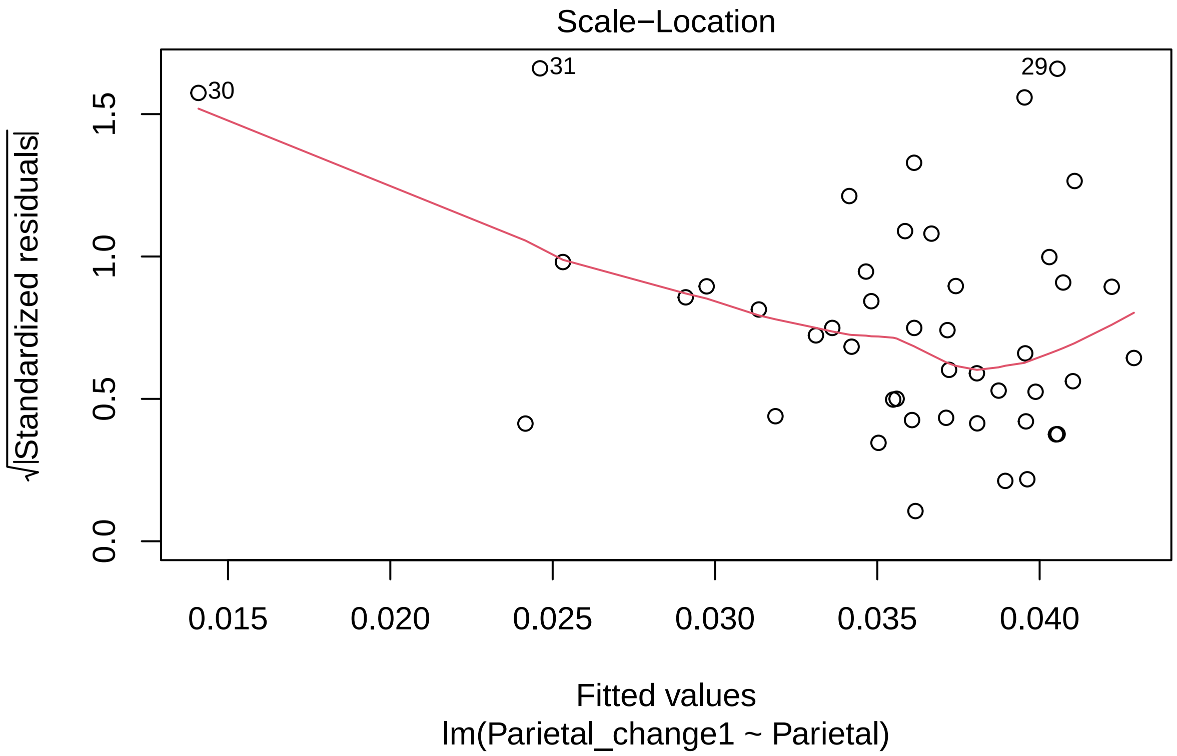
```
plot(y = filename$Parietal_change1, x = filename$Central)
```

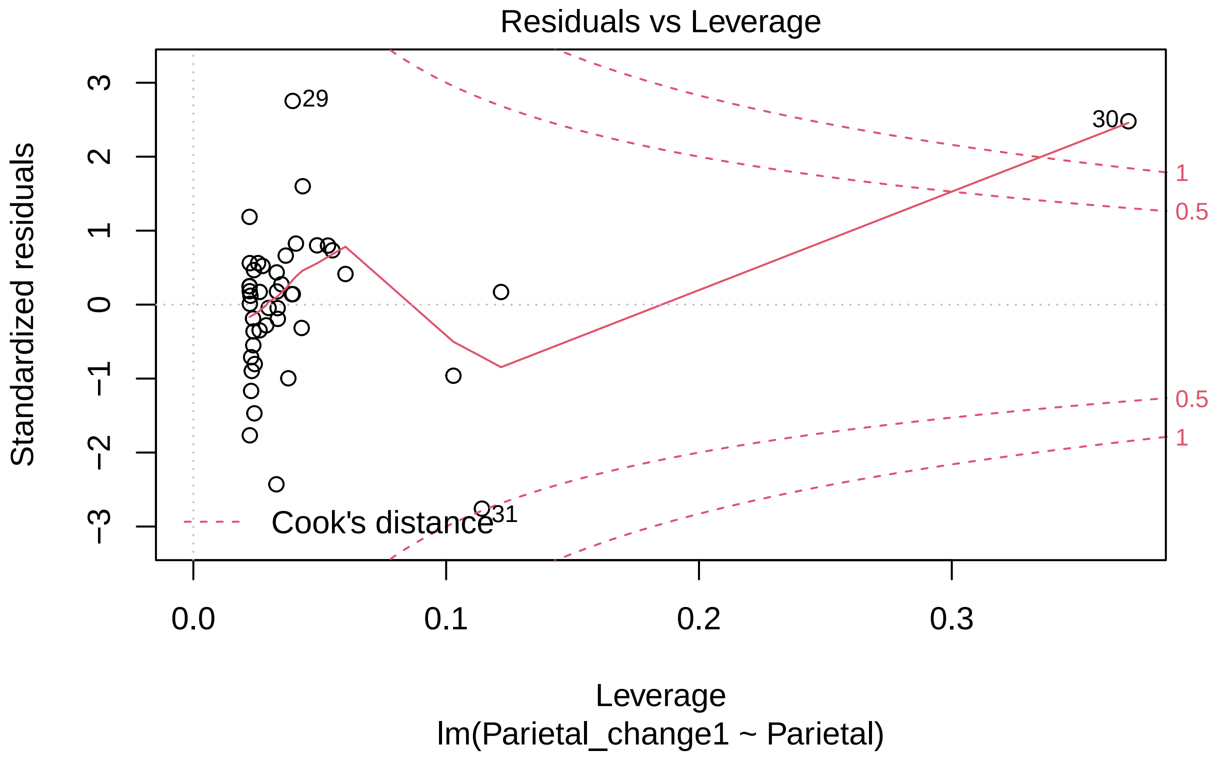


```
model1 <- lm(formula = Parietal_change1 ~ Parietal, data = filename )
plot(model1)
```


Residuals vs Fitted
Residuals
2
1
0
−1
−2
29
31
43
0.015
0.020
0.025
0.030
0.035
0.040
Fitted values
lm(Parietal_change1 ~ Parietal)


Normal Q−Q
3
2
1
0
−1
−2
Standardized residuals
29
30
31
−2
−1
0
1
2
Theoretical Quantiles
lm(Parietal_change1 ~ Parietal)


Scale−Location
30
31
29
√|Standardized residuals|
0.0
0.5
1.0
1.5
0.015
0.020
0.025
0.030
0.035
0.040
Fitted values
lm(Parietal_change1 ~ Parietal)

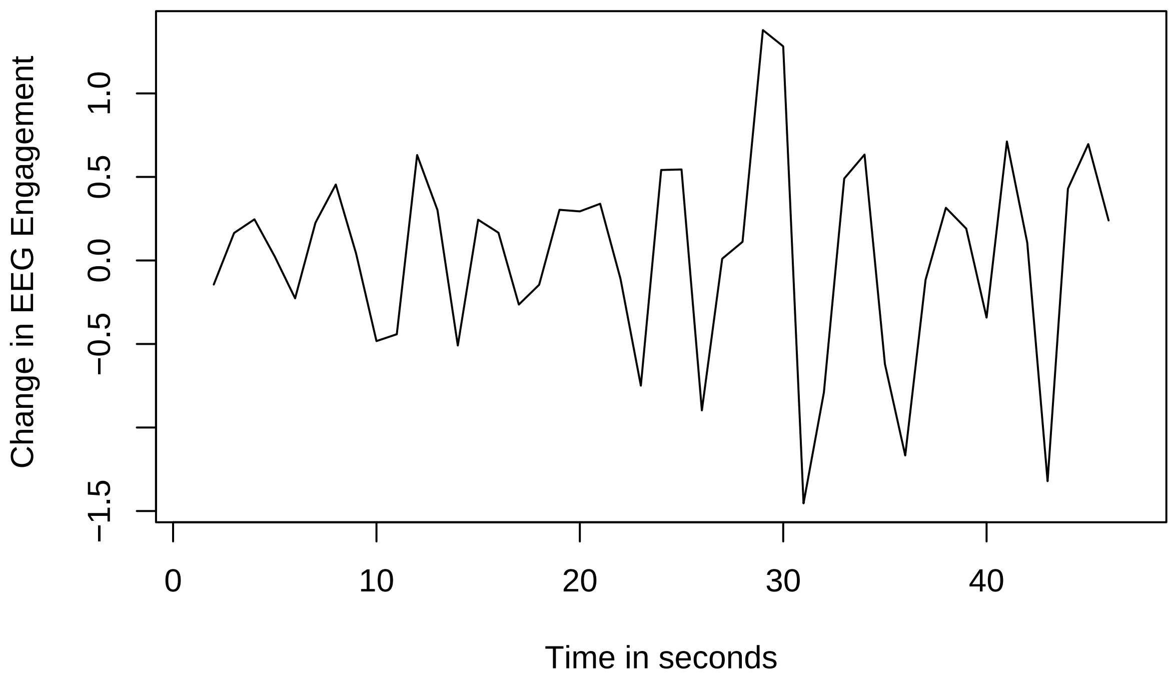


```
summary(model1)
```

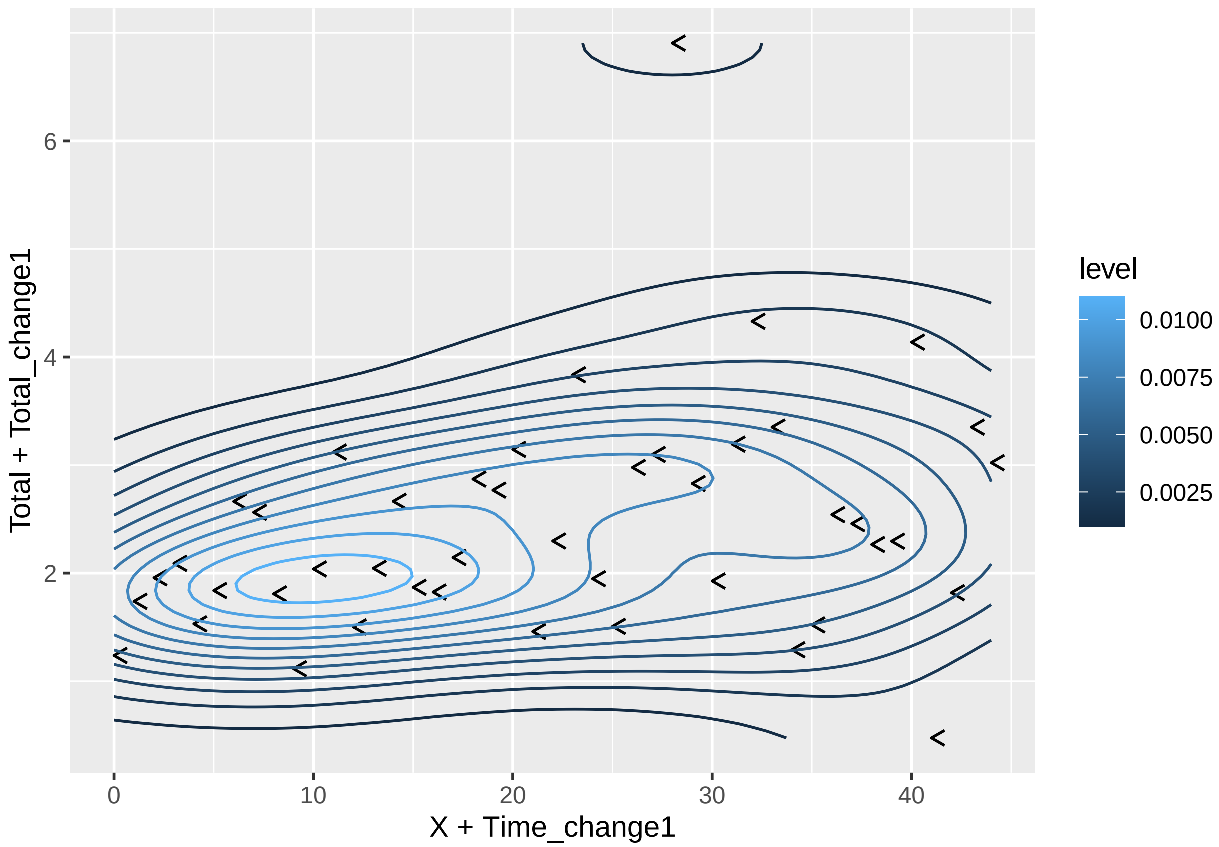
```
##
## Call:
## lm(formula = Parietal_change1 ~ Parietal, data = filename)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9424 -0.2678  0.1037  0.3857  2.0195
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.049547   0.302423   0.164    0.871
## Parietal    -0.005361   0.109197  -0.049    0.961
##
## Residual standard error: 0.7483 on 43 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  5.604e-05,  Adjusted R-squared:  -0.0232
## F-statistic: 0.00241 on 1 and 43 DF,  p-value: 0.9611
```

Plotting all regions

```
plot(filename$Total_change1,type='l', xlab = "Time in seconds", ylab =" Change in EEG Engagement")
```

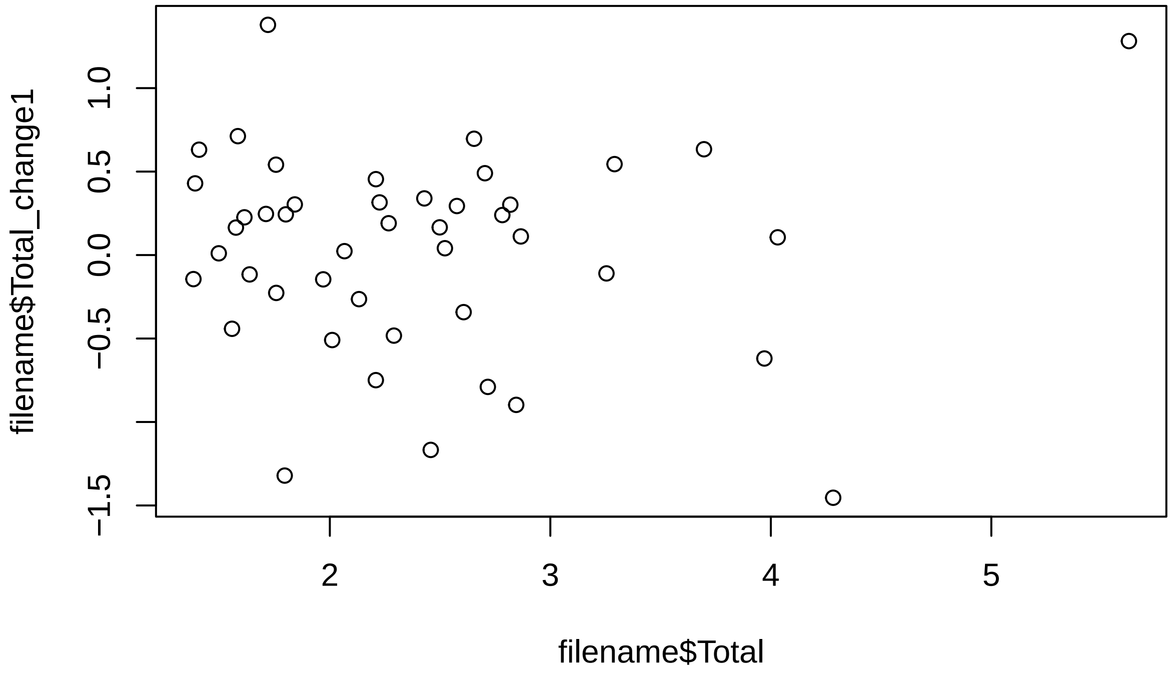


```
ggplot(data=filename,aes(x=X + Time_change1,y=Total + Total_change1))+
  geom_segment(aes(xend=X + Time_change1,yend=Total + Total_change1),arrow=arrow(length=unit(.2,"cm")))
  stat_density2d(aes(colour=..level..))+
  labs(list(title="Participant 1 Vector Density Plot", x= "Time", y = "EEG Engagement"))
```

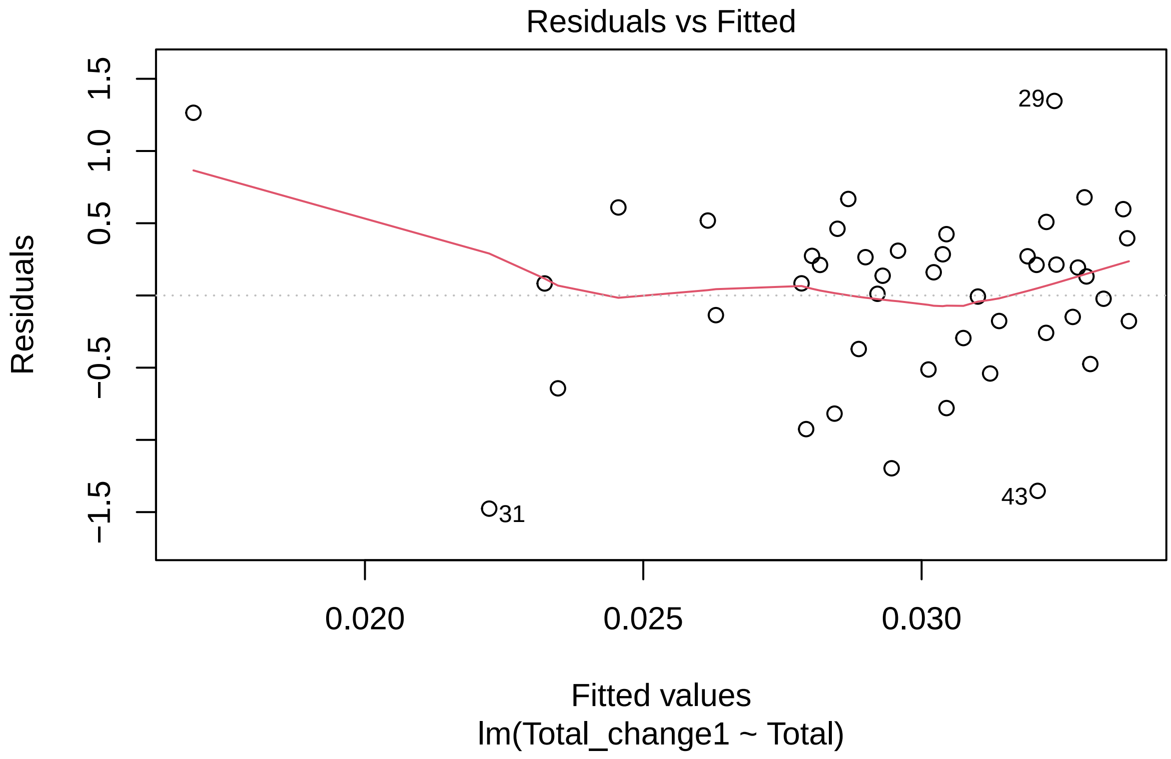


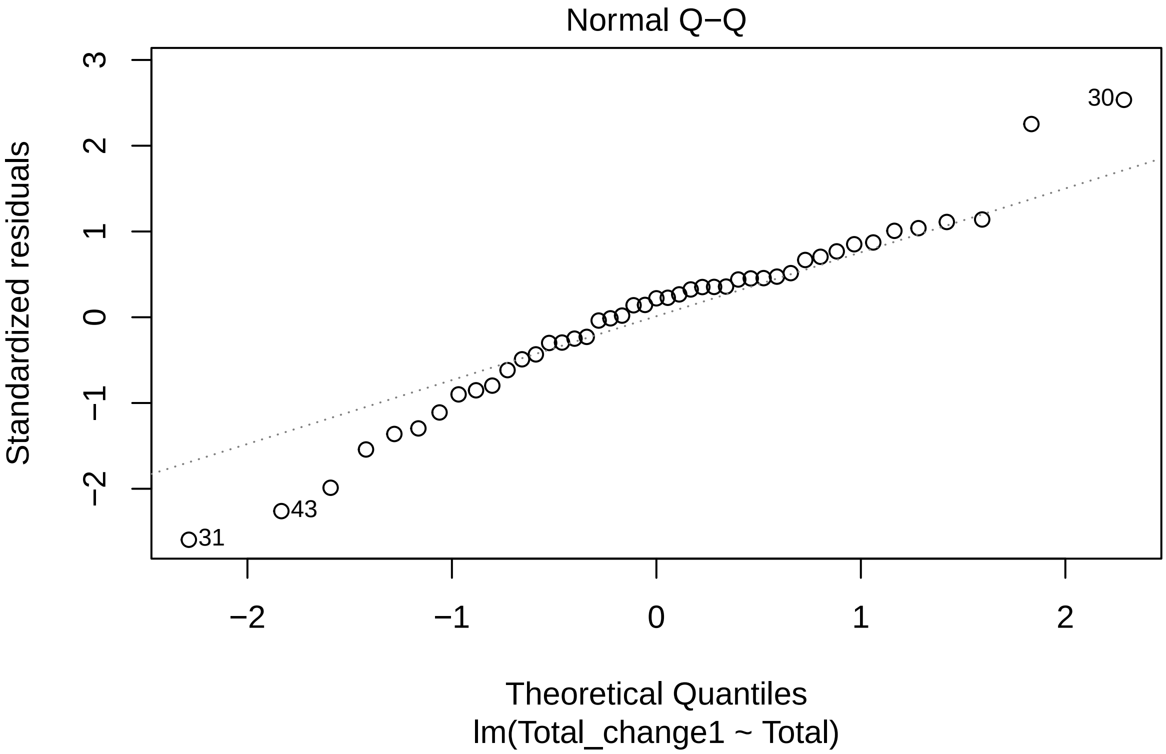
Plotting the linear predictor for all regions

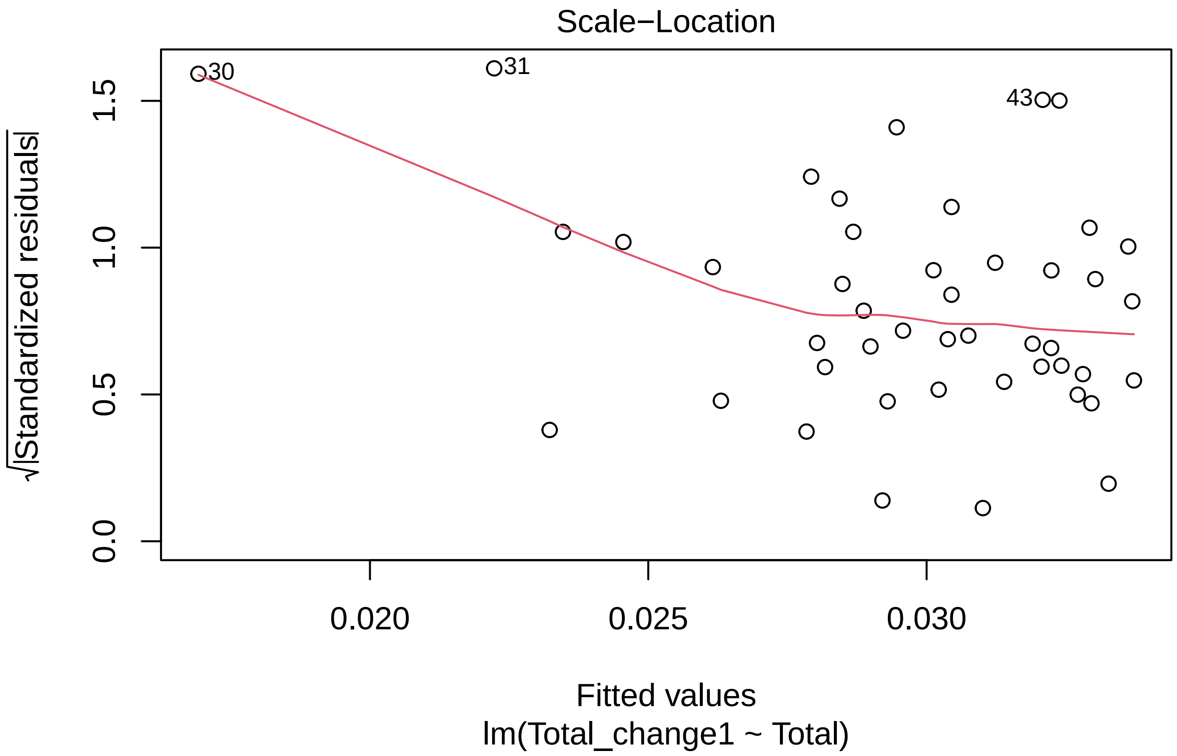
```
plot(y = filename$Total_change1, x = filename$Total)
```

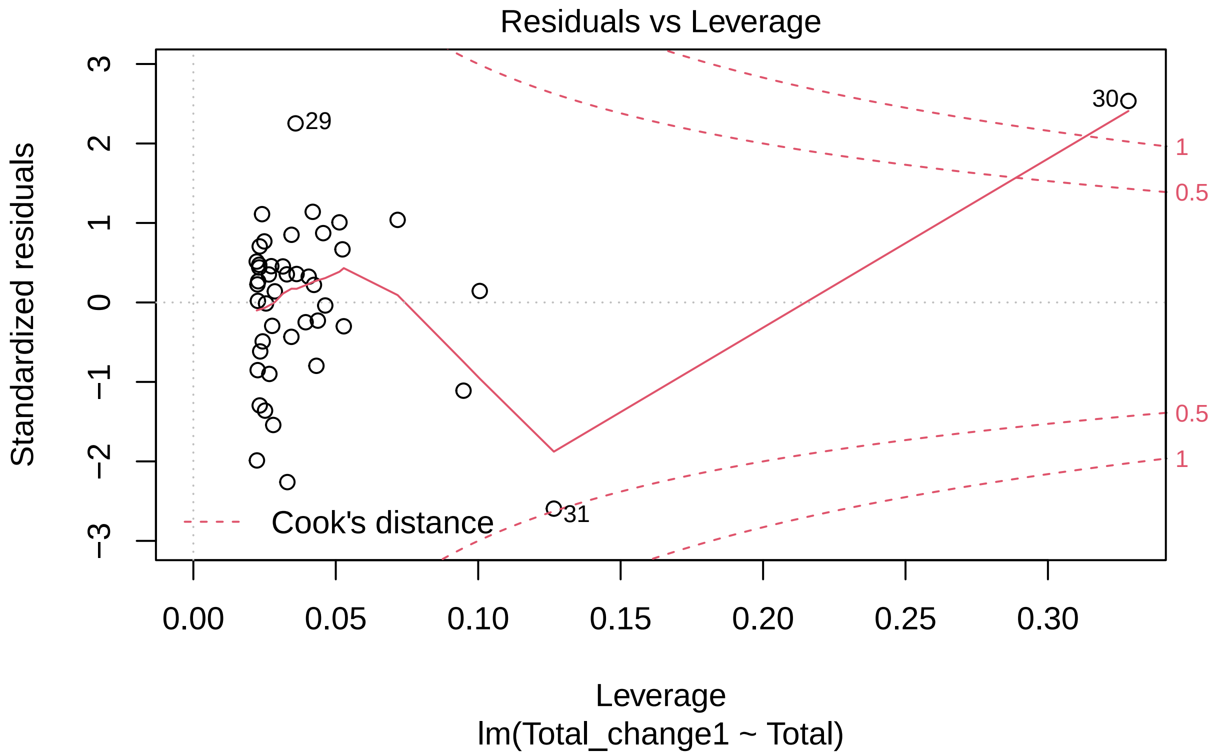


```
model1 <- lm(formula = Total_change1 ~ Total, data = filename )
plot(model1)
```


Residuals vs Fitted
Residuals
1.5
1.0
0.5
−0.5
−1.5
29
31
43
0.020
0.025
0.030
Fitted values
lm(Total_change1 ~ Total)


Normal Q−Q
Standardized residuals
3
2
1
0
−1
−2
−2
−1
0
1
2
30
31
43
Theoretical Quantiles
lm(Total_change1 ~ Total)


Scale−Location
30
31
43
1.5
1.0
0.5
0.0
√|Standardized residuals|
0.020
0.025
0.030
Fitted values
lm(Total_change1 ~ Total)



```
summary(model1)
```

```
##
## Call:
## lm(formula = Total_change1 ~ Total, data = filename)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4760 -0.2949  0.1315  0.3096  1.3466
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.039196   0.266695   0.147    0.884
## Total       -0.003961   0.104471  -0.038    0.970
##
## Residual standard error: 0.6088 on 43 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  3.343e-05,  Adjusted R-squared:  -0.02322
## F-statistic: 0.001437 on 1 and 43 DF,  p-value: 0.9699
```

Bartels rank tests per region

```r
bartels.rank.test(filename$Frontal, alternative = "two.sided")
```

```
##
##  Bartels Ratio Test
##
## data:  filename$Frontal
## statistic = -1.0639, n = 47, p-value = 0.2874
## alternative hypothesis: nonrandomness
```

```r
bartels.rank.test(filename$Central, alternative = "two.sided")
```

```
##
##  Bartels Ratio Test
##
## data:  filename$Central
## statistic = -0.99509, n = 47, p-value = 0.3197
## alternative hypothesis: nonrandomness
```

```r
bartels.rank.test(filename$Parietal, alternative = "two.sided")
```

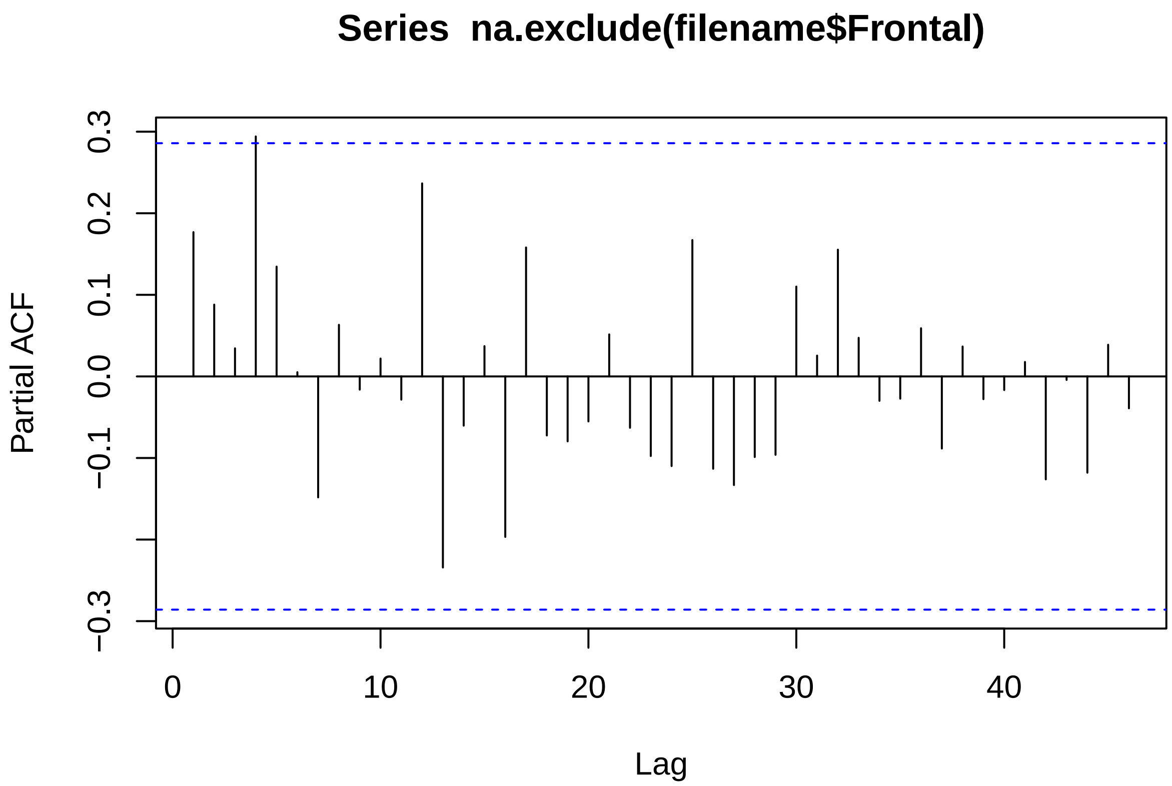
```
##
##  Bartels Ratio Test
##
## data:  filename$Parietal
## statistic = -1.5558, n = 47, p-value = 0.1197
## alternative hypothesis: nonrandomness
```

```r
bartels.rank.test(filename$Total, alternative = "two.sided")
```

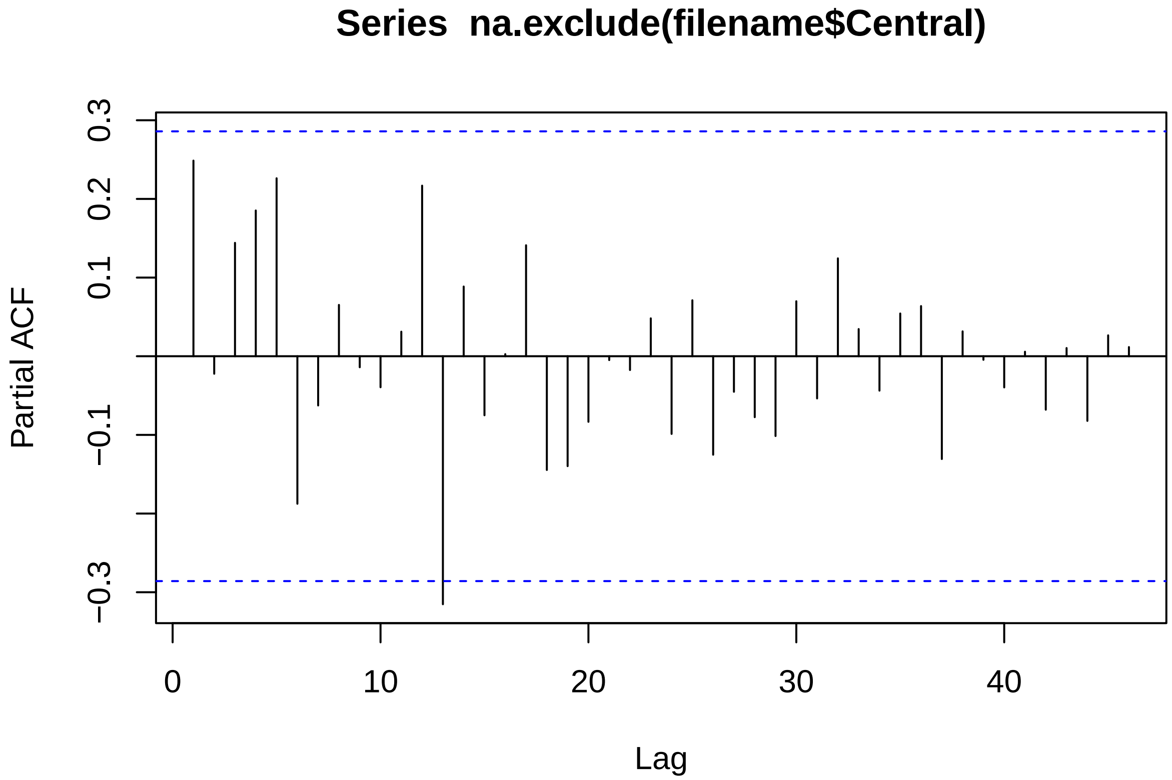
```
##
##  Bartels Ratio Test
##
## data:  filename$Total
## statistic = -1.1227, n = 47, p-value = 0.2616
## alternative hypothesis: nonrandomness
```

Calculating auto-correlation for each region

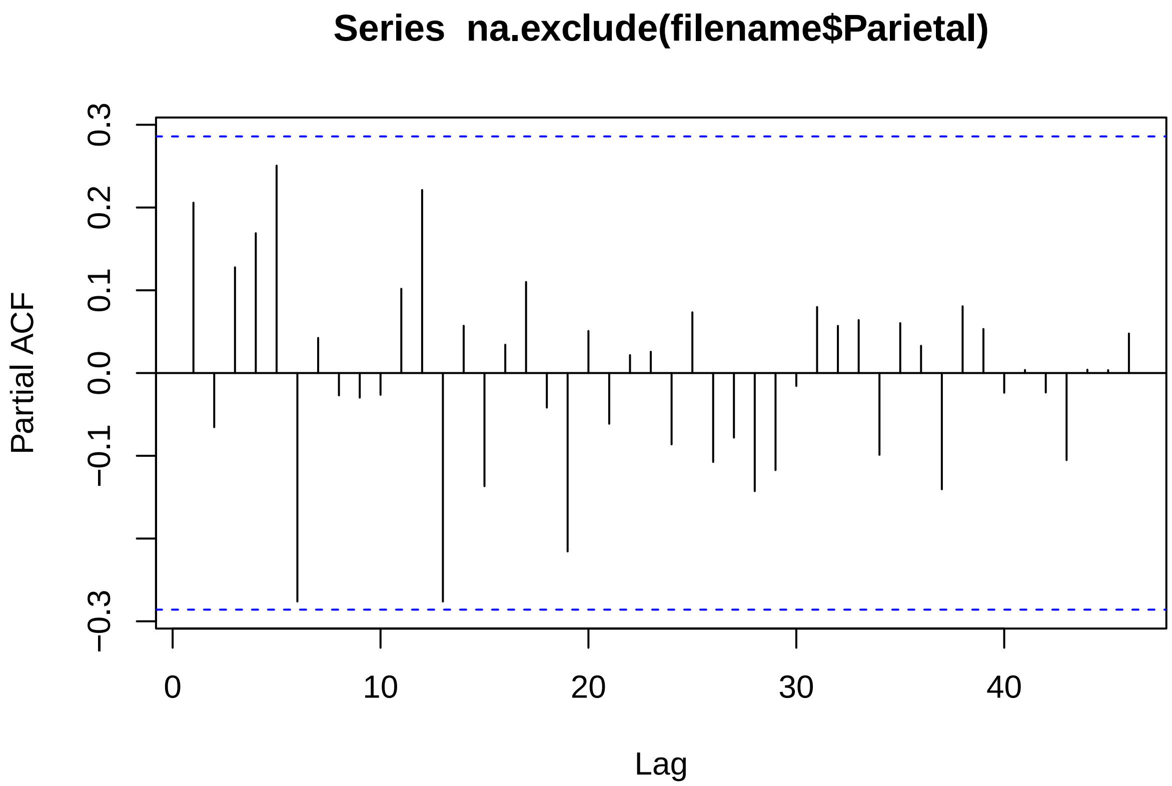
```r
pacf(na.exclude(filename$Frontal),lag.max = 1000)
```



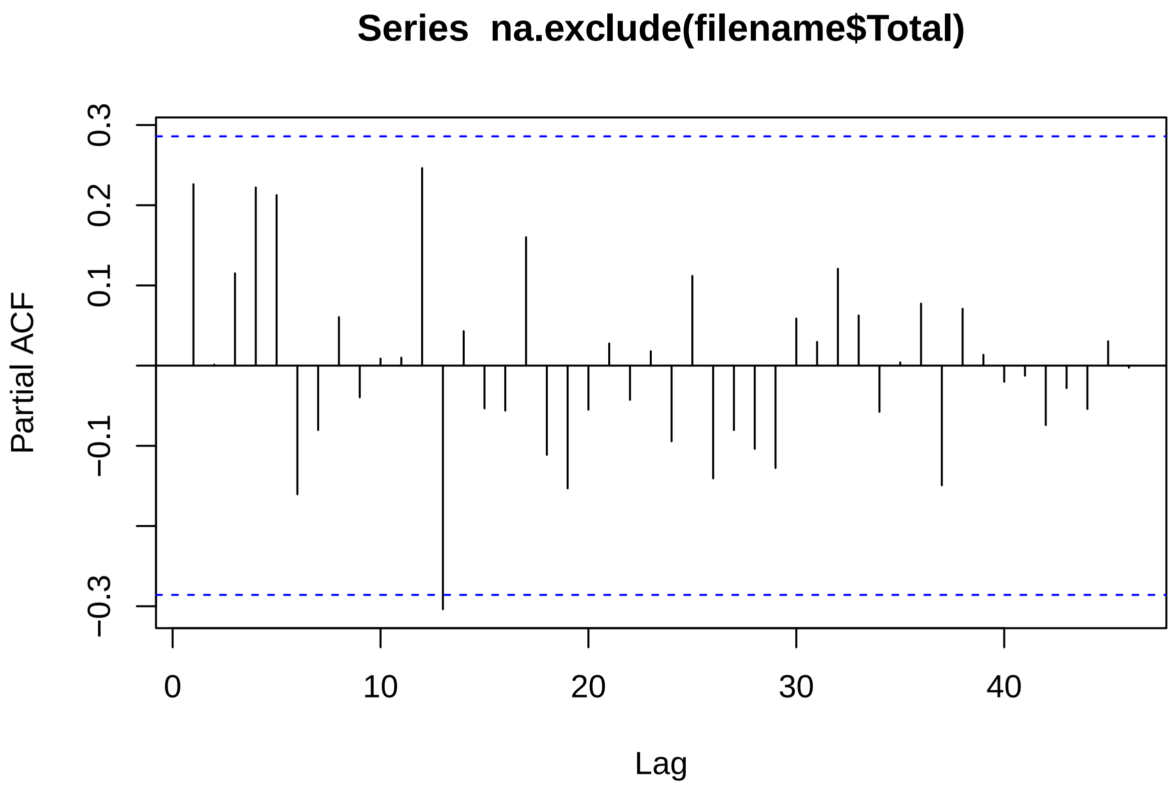
```
pacf(na.exclude(filename$Central),lag.max = 1000)
```



```
pacf(na.exclude(filename$Parietal),lag.max = 1000)
```



```
pacf(na.exclude(filename$Total),lag.max = 1000)
```



KPSS tests for each region

```
kpss.test(na.exclude(filename$Frontal), lshort=TRUE,
          null="Level")
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  na.exclude(filename$Frontal)
## KPSS Level = 0.62314, Truncation lag parameter = 3, p-value = 0.02053
```

```
kpss.test(na.exclude(filename$Central), lshort=TRUE,
          null="Level")
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  na.exclude(filename$Central)
## KPSS Level = 0.53171, Truncation lag parameter = 3, p-value = 0.03452
```

```
kpss.test(na.exclude(filename$Parietal), lshort=TRUE,
          null="Level")
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  na.exclude(filename$Parietal)
## KPSS Level = 0.479, Truncation lag parameter = 3, p-value = 0.0464
```

```
kpss.test(na.exclude(filename$Total), lshort=TRUE,
          null="Level")
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  na.exclude(filename$Total)
## KPSS Level = 0.55675, Truncation lag parameter = 3, p-value = 0.02889
```

Change point analysis per region

Frontal

```
ts <- matrix(na.exclude(filename$Frontal))

e.out <- ecp::e.divisive(ts, R=500, sig.lvl=.002)
df.e <- length(which(e.out$p.values<.002))
e.out$estimates
```

```
## [1]  1 48
```

```
df.e
```

```
## [1] 0
```

Central

```
ts <- matrix(na.exclude(filename$Central))

e.out <- ecp::e.divisive(ts, R=500, sig.lvl=.002)
df.e <- length(which(e.out$p.values<.002))
e.out$estimates
```

```
## [1]  1 48
```

```
df.e
```

```
## [1] 0
```

Parietal

```
ts <- matrix(na.exclude(filename$Parietal))

e.out <- ecp::e.divisive(ts, R=500, sig.lvl=.002)
df.e <- length(which(e.out$p.values<.002))
e.out$estimates
```

```
## [1]  1 48
```

```
df.e
```

```
## [1] 0
```

Total

```
ts <- matrix(na.exclude(filename$Total))

e.out <- ecp::e.divisive(ts, R=500, sig.lvl=.002)
df.e <- length(which(e.out$p.values<.002))
e.out$estimates
```

```
## [1]  1 48
```

```
df.e
```

```
## [1] 0
```